

Introduction to Cloud Computing

What is Cloud?

Introduction to Cloud Computing

In simple words, Cloud computing is – Placing your data on someone else's datacenter, letting them manage underline hardware Infrastructure (optionally underline Database or applications too); while having your full control on the data, and accessing that data through Internet or dedicated network.

Cloud computing is a model for enabling **universal, on-demand** access to a **shared pool** of configurable computing resources (e.g., computer networks, servers, storage, applications and services), which can be **rapidly provisioned** and **released** with **minimal management effort** on **Pay-per-use basis**.

Free available cloud Examples:

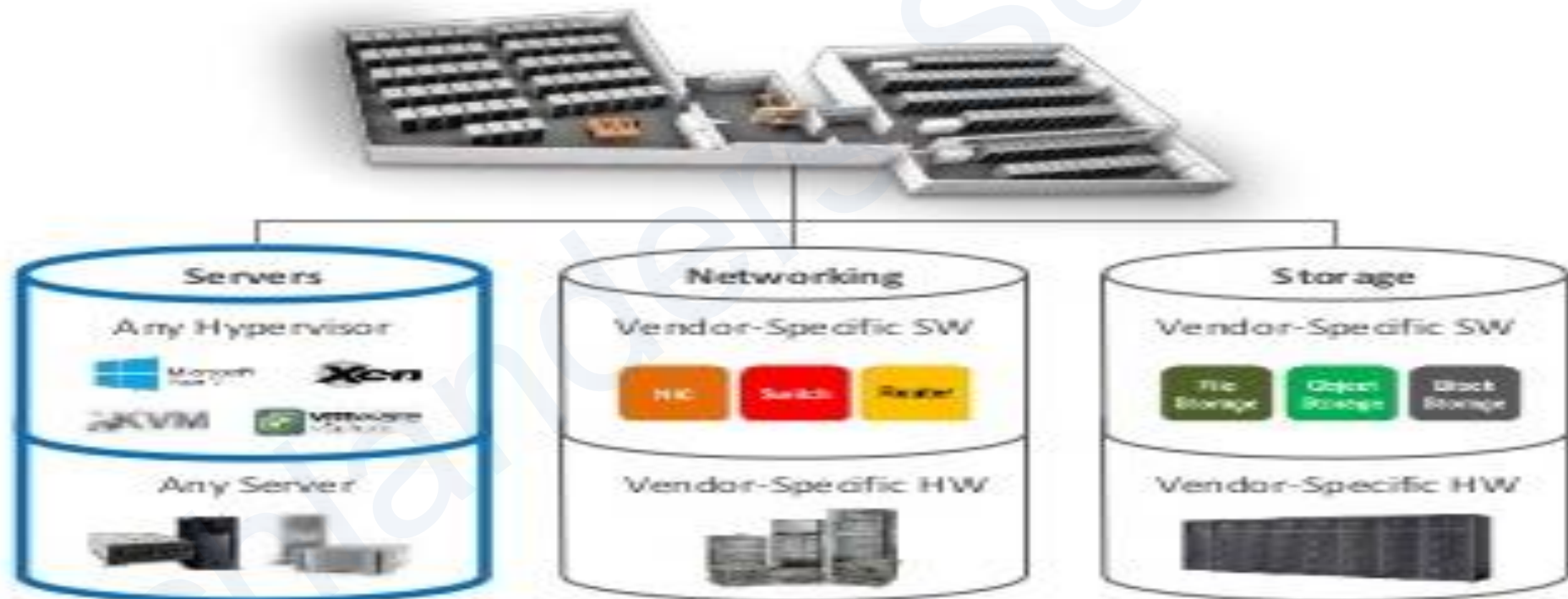
Paid available cloud Examples:

Gmail, IRCTC, WhatsApp/Facebook

AWS, Azure(Microsoft), Oracle Cloud


Traditional DataCenters

Traditional Data Center




Traditional DataCenters

Main issues with Traditional IT Infrastructure.

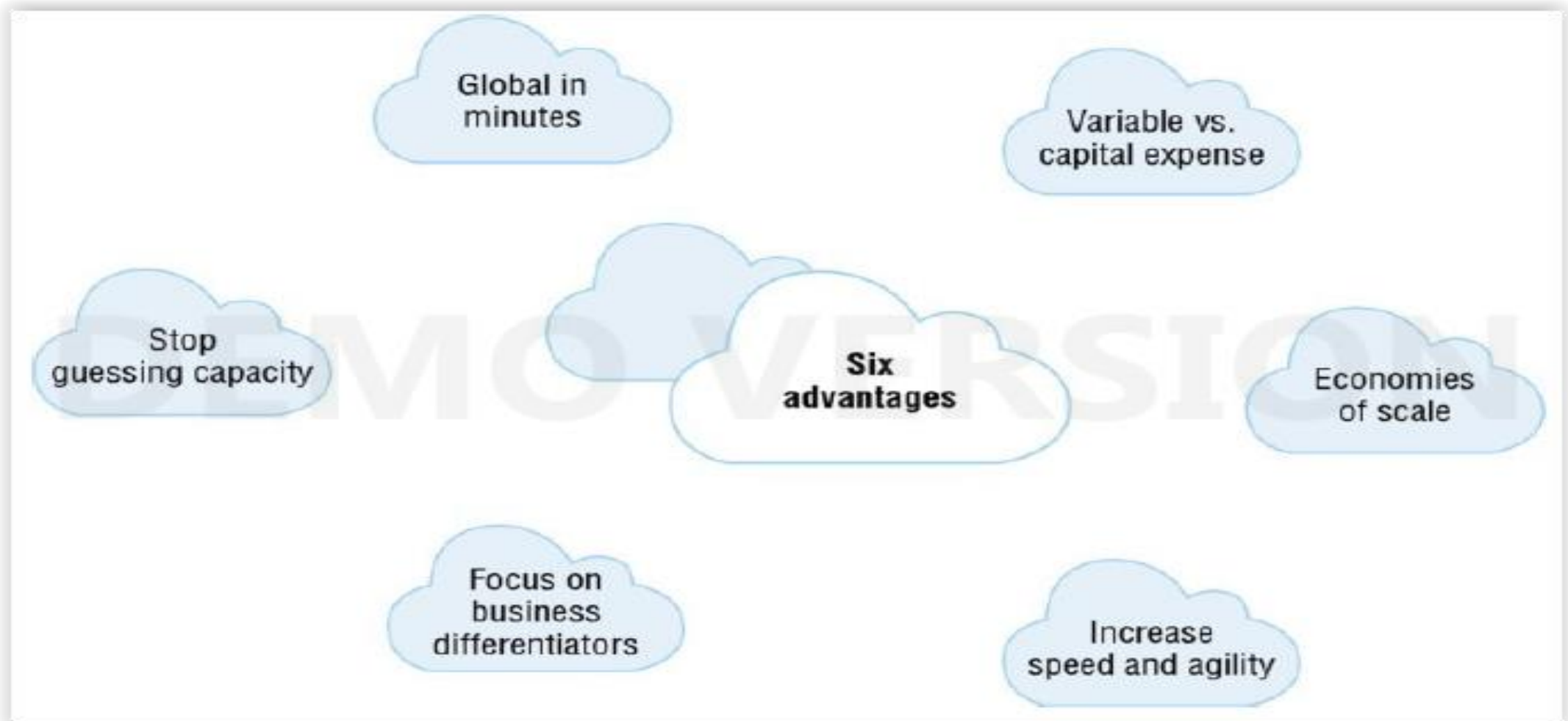
- Infrastructure is not a core business
 - Hard to Scale
 - Dedicated Infrastructure teams
 - Dedicated Datacenters
 - Dependency on vendors (servers, switches, cables etc.)
 - Underutilized Resources
 - High Cost
 - Difficult Capacity Planning
 - On-Spot demands were hard to manage
 - Provisioning resources was very time consuming
- 

Why cloud?

To overcome all of the discussed challenges, IT infrastructure domain drifted towards Service based model which is a real “cloud computing”

- No Dedicated Datacenter
 - No Different Infrastructure Teams
 - Higher/Faster Scalability
 - Elasticity
 - Pay per use model
 - Option to adopt high availability
 - Better performance
 - Instant provisioning
 - Optimized use of resources
 - On demand scaling to any extent
 - No to worry about capacity planning
- 

Cloud Advantages



Cloud Service Models

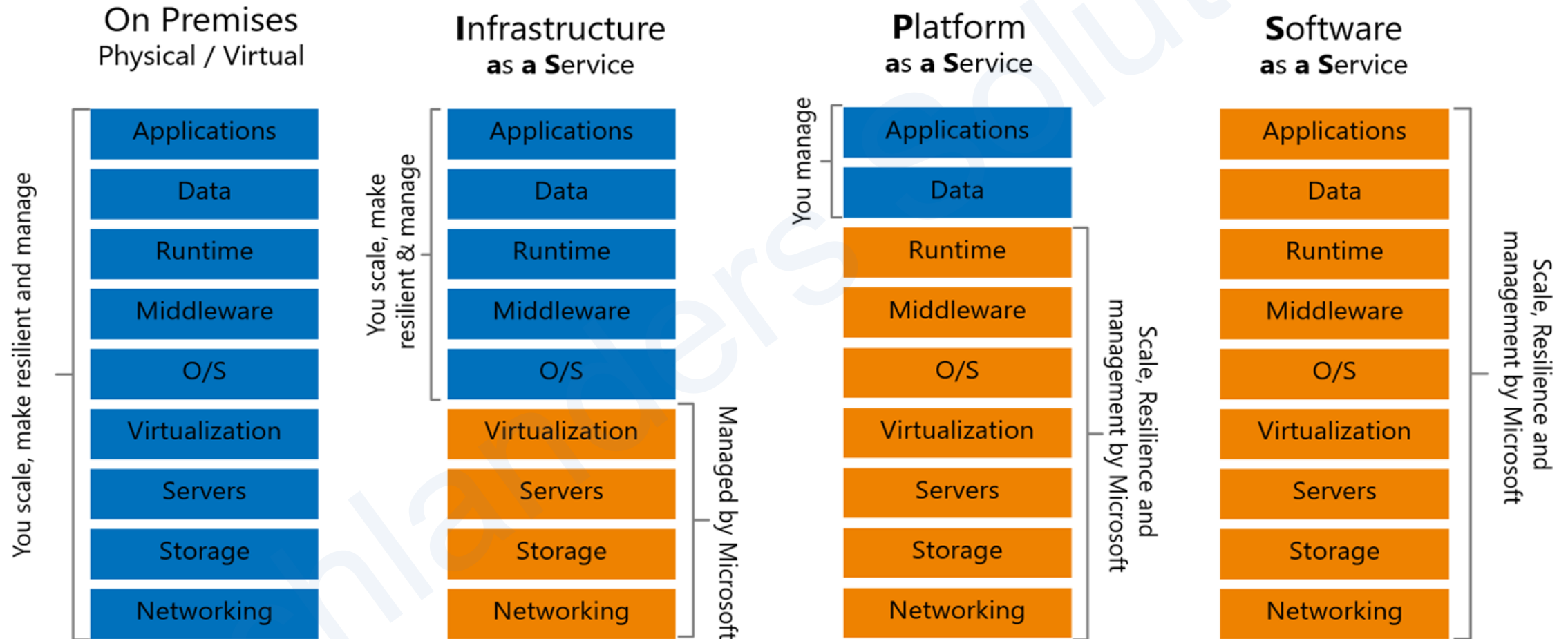
There are three Cloud Computing Service Models:

Infrastructure as a Service (IaaS)

Platform as a Service (PaaS)

Software as a Service (SaaS)

Responsibility- Who owns What?



Responsibility- Who owns What?

Pizza as a Service



Cloud Service Models - IaaS

IaaS is the most basic Cloud Service Model

It offers Underline Infrastructure for Compute, Storage and Networking

Infrastructure can be selected by customers as per their choice and Pay-per-use model.

Examples: Bare metal servers, virtual Instances, Load balancers



IaaS - Benefits

Drastic reduction in capital investment

Easily Scalable

Pay only for the used resources

High Flexibility

Reduced infrastructure support teams

Cloud Service Models - PaaS

Another service model, where cloud provider manages the OS & middleware part, along with IaaS

Provide capability to deploy applications on cloud infrastructure without managing underline Infra

Consumers are responsible for managing deployed applications and their environment specific configurations

Examples: web servers and databases



PaaS - Benefits

Includes all IaaS benefits

No upfront licensing cost

More reduction in Infrastructure support team

Rapid time to market



Cloud Service Models - SaaS

SaaS deliver complete application to the consumers over the internet.

Consumers are not responsible for managing any application or underlying infrastructure.

SaaS application are delivered as “one-to-many” model.

Examples: office365, Gmail, WhatsApp, JIRA, GIT, Service Now

SaaS - Benefits

Includes all discussed benefits which we get in PaaS

Ability to access from anywhere

Ability to access from multiple devices

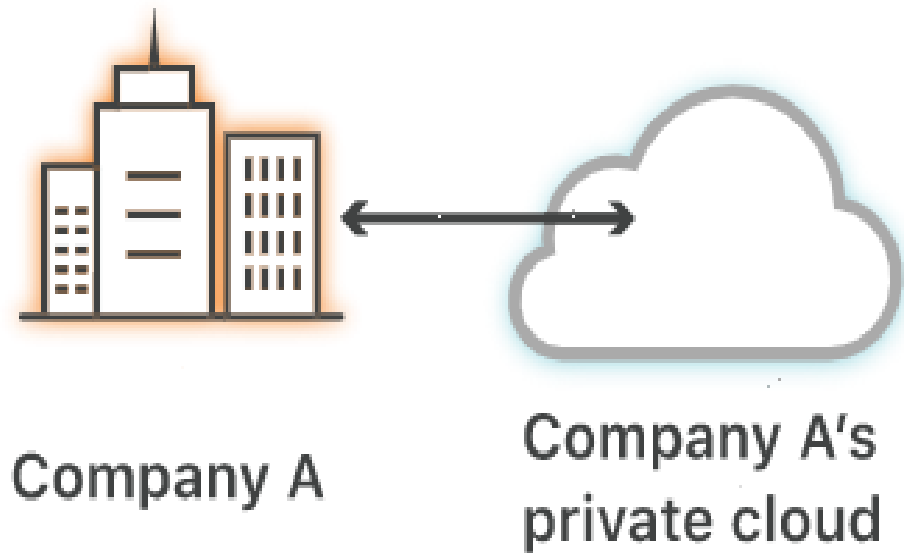
No installations and maintenance requirements

No Application management/Licensing Required

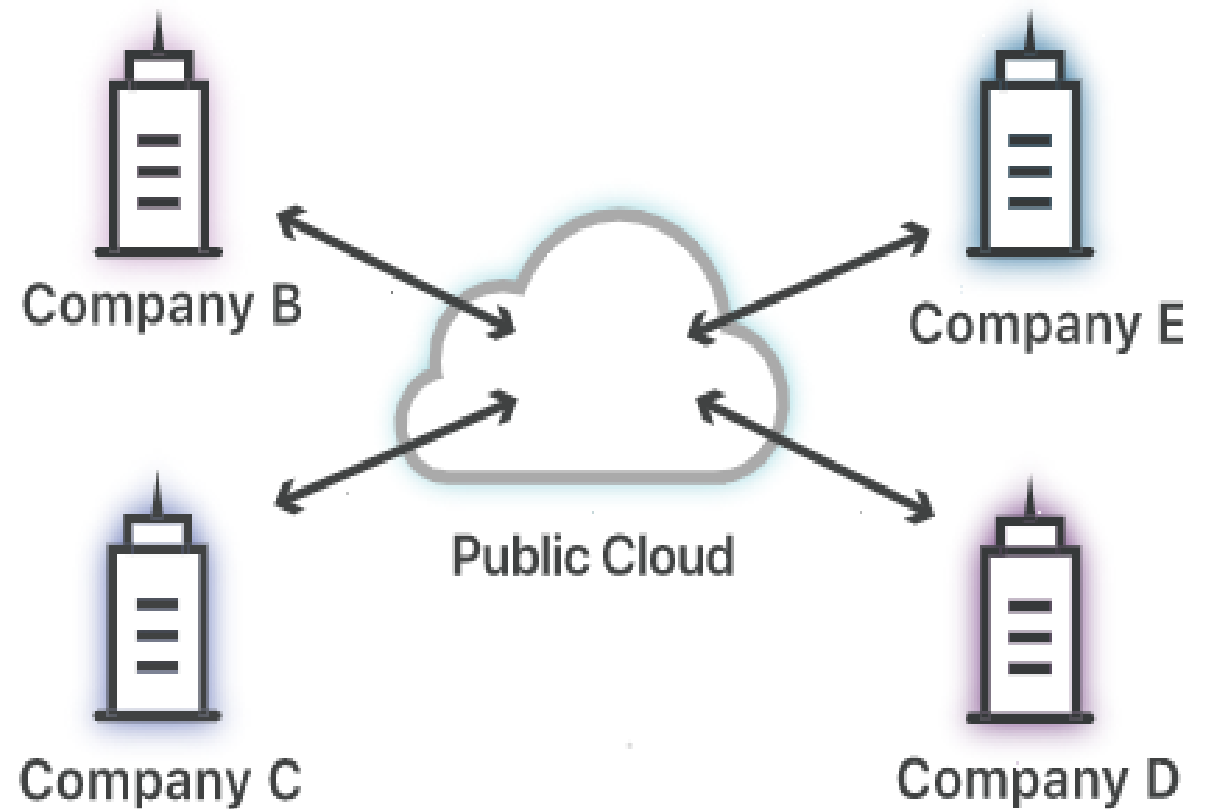
Cloud Essentials Characteristics



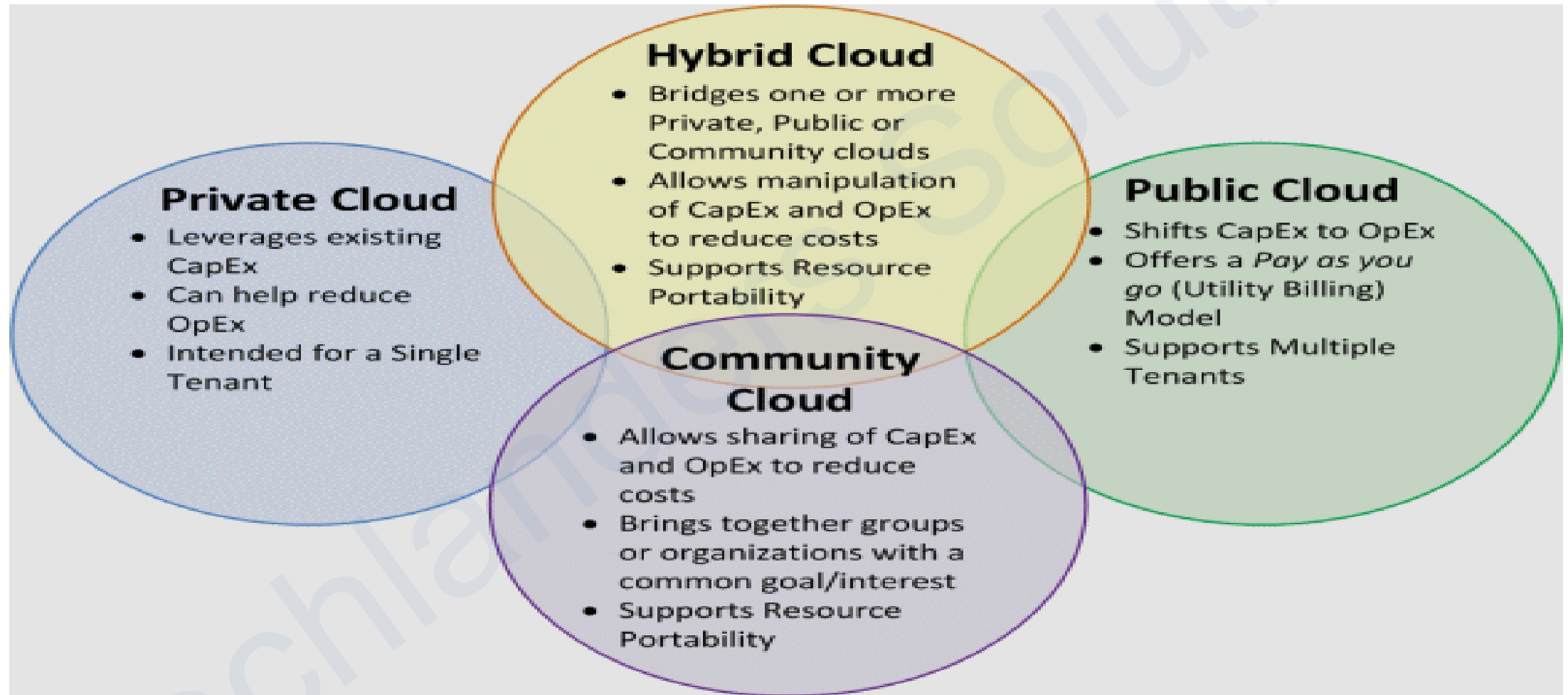
Private cloud



Public cloud shared by multiple companies



Cloud Deployments Types



Cloud's Major Use Cases



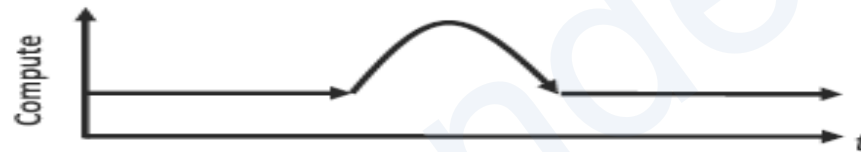
On and Off

On and off workloads (e.g. batch job)
Over provisioned capacity is wasted
Time to market can be cumbersome



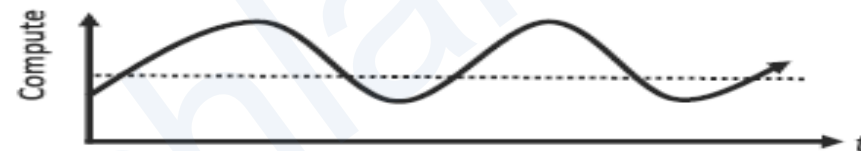
Growing Fast

Successful services need to grow/scale
Keeping up with growth is a big IT challenge
Cannot provision hardware fast enough



Unpredictable Bursting

Unexpected/unplanned peak in demand
Sudden spike impacts performance
Cannot over provision for extreme cases

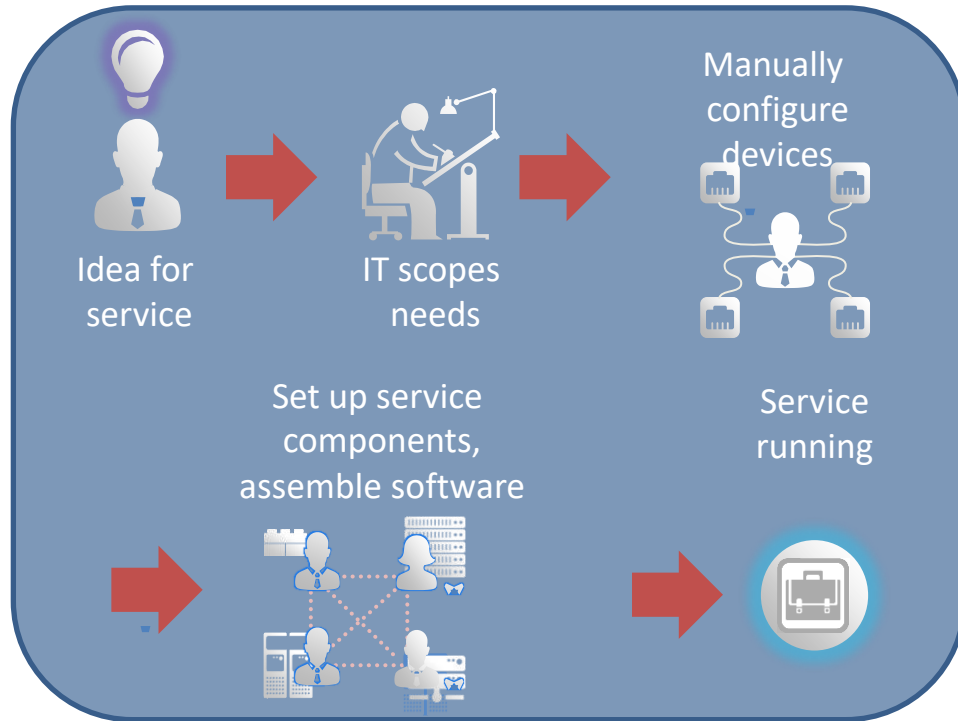


Predictable Bursting

Services with micro seasonality trends
Peaks due to periodic increased demand
IT complexity and wasted capacity

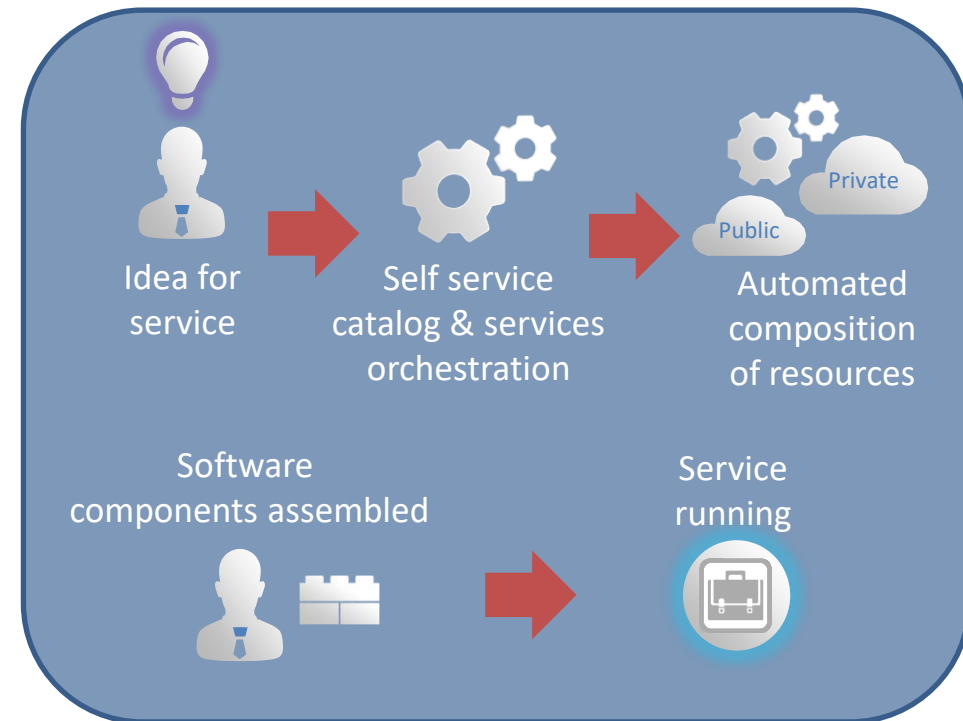
Business Impact of Cloud

Traditional Datacenter



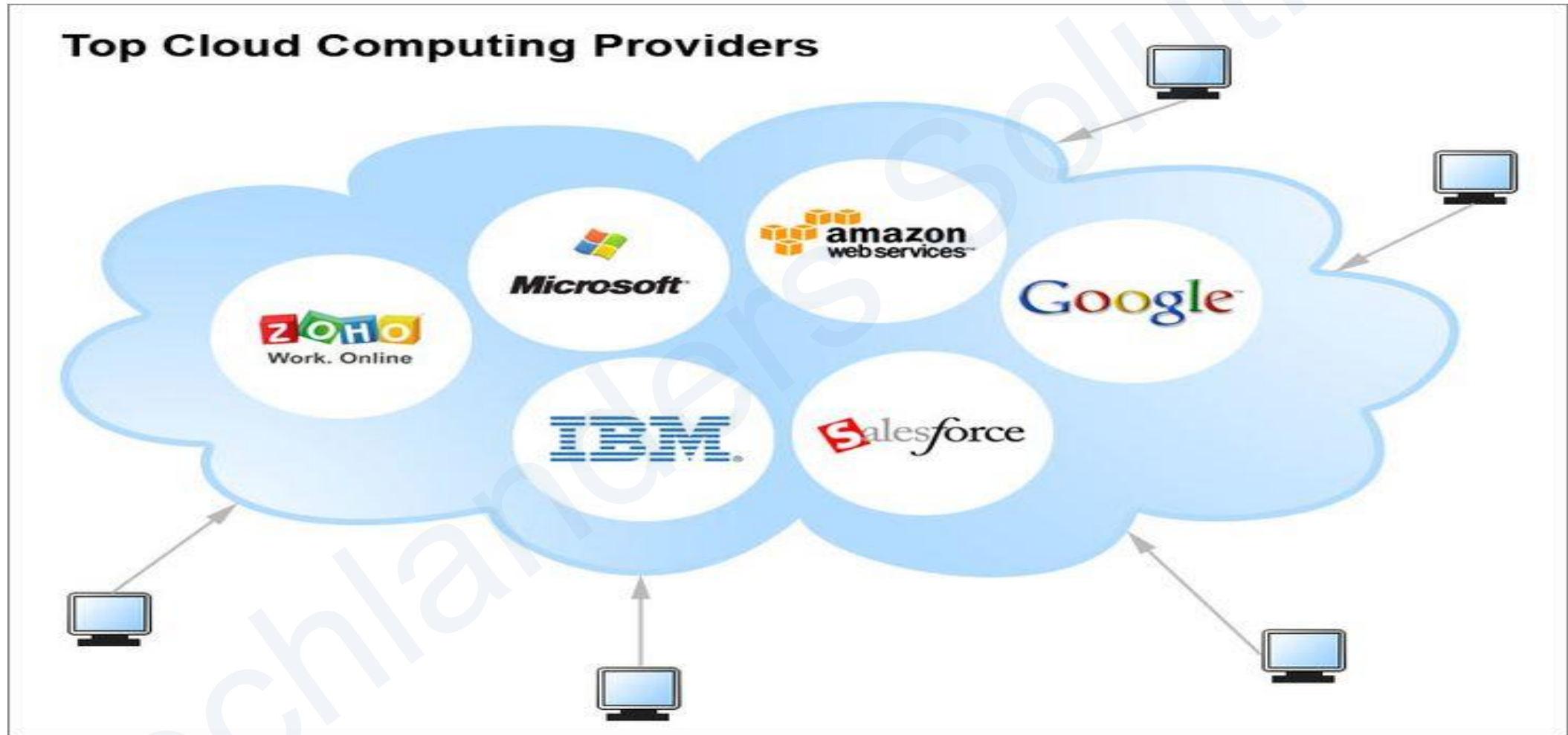
Time to Provision New Service: Months

Cloud Infrastructure

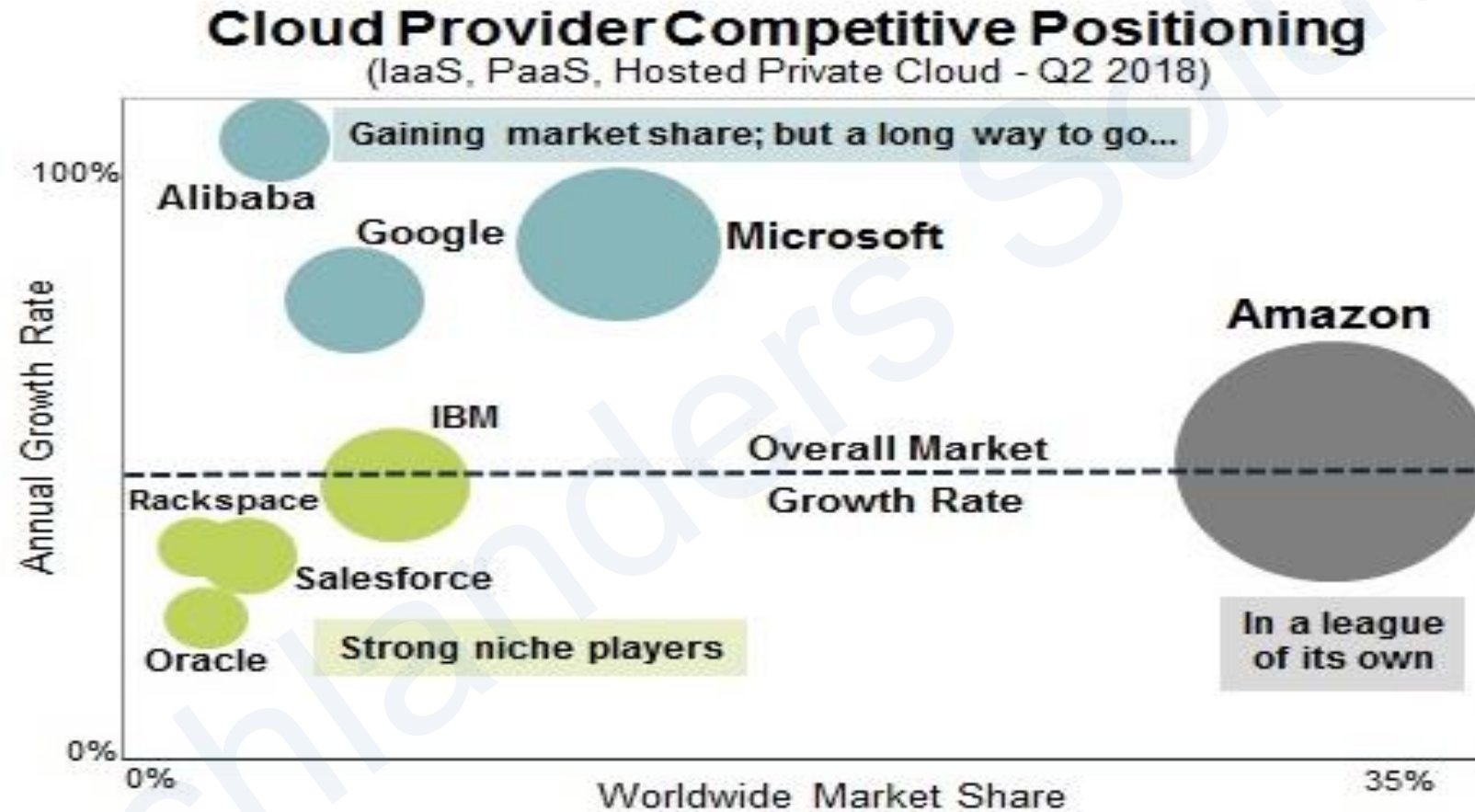


Time to Provision New Service: Minutes

Major Cloud Vendors

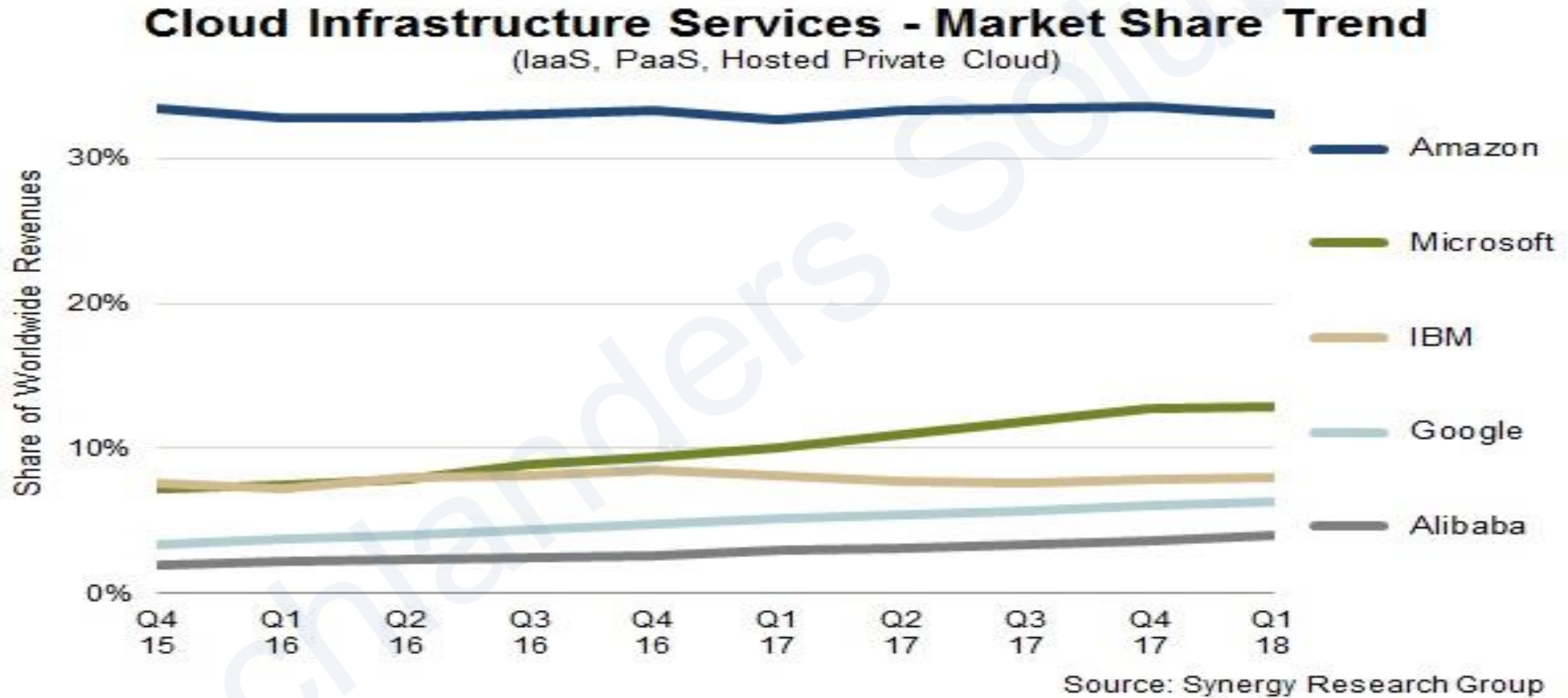


Who stands where?



Source: Synergy Research Group

Who stands where?



Knowledge Checks

- Which Service Level (IaaS, PaaS, SaaS) provides you most control?
- What is Hybrid Cloud?
- Can two public clouds be connected?
- Connecting two public clouds, will be know as public cloud or Hybrid?
- Cloud provided Database, is a PaaS or SaaS?

AWS (Amazon Cloud)

Amazon Web Services

AWS (Amazon Web Services) is a group of web services (also known as cloud services) being provided by Amazon since 2006.

AWS provides huge list of services starting from basic IT infrastructure like CPU, Storage as a service, to advance services like Database as a service, Serverless applications, IOT, Machine Learning services etc..

Hundreds of instances can be build and use in few minutes as and when required, which saves ample amount of hardware cost for any organizations and make them efficient to focus on their core business areas.

Currently AWS is present and providing cloud services in more than 190 countries.

Well-known for IaaS, but now growing fast in PaaS and SaaS.



Why AWS?

Low Cost: AWS offers, pay as you go pricing. AWS models are usually cheapest among other service providers in the market.


Instant Elasticity: You need 1 server or 1000's of servers, AWS has a massive infrastructure at backend to serve almost any kind of infrastructure demands, with pay for what you use policy.

Scalability: Facing some resource issues, no problem within seconds you can scale up the resources and improve your application performance. This cannot be compared with traditional IT datacenters.

Multiple OS's: Choice and use any supported Operating systems.

Multiple Storage Options: Choice of high I/O storage, low cost storage. All is available in AWS, use and pay what you want to use with almost any scalability.

Secure: AWS is PCI DSS Level1, ISO 27001, FISMA Moderate, HIPAA, SAS 70 Type II passed. In-fact systems based on AWS are usually more secure than in-house IT infrastructure systems.



AWS Global Infrastructure

AWS Regions:

- Geographic Locations
- Consists of at least two Availability Zones(AZs)
- All of the regions are completely independent of each other with separate Power Sources, Cooling and Internet connectivity.

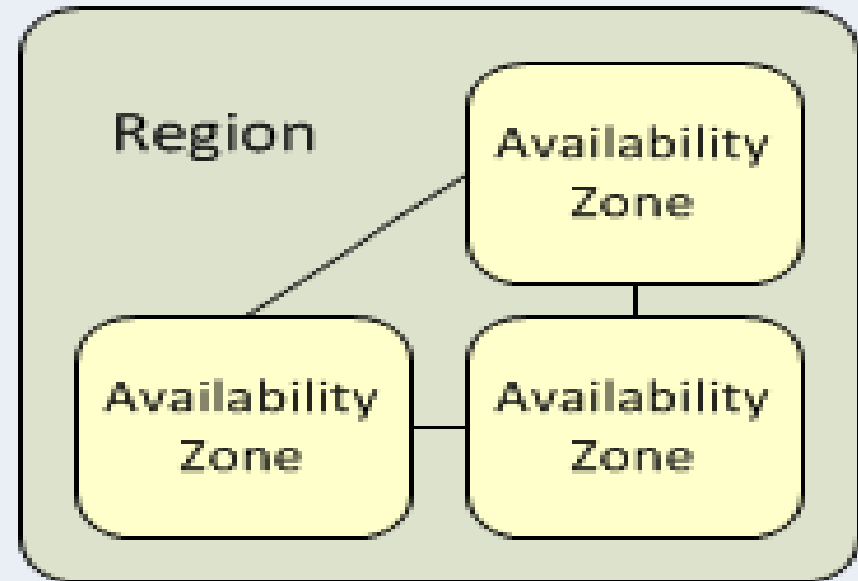
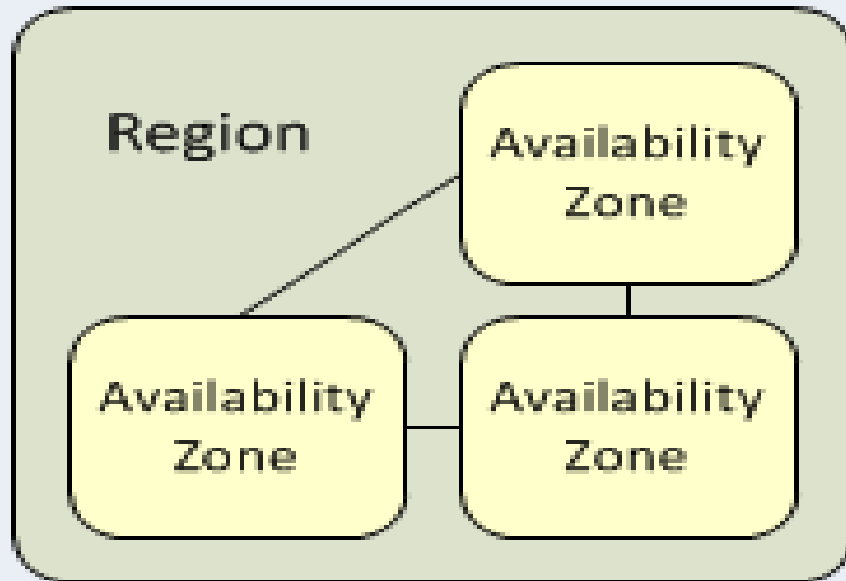
AWS Availability Zones

- AZ is a distinct location within a region
- Each zone is insulated (with low-latency links) from other to support single point of failures
- Each Region has minimum two AZ's
- Most of the services/resources are replicated across AZs for HA/DR purpose.

Note: Resources aren't replicated across regions unless you do so specifically.

AWS Global Infrastructure

Amazon Web Services



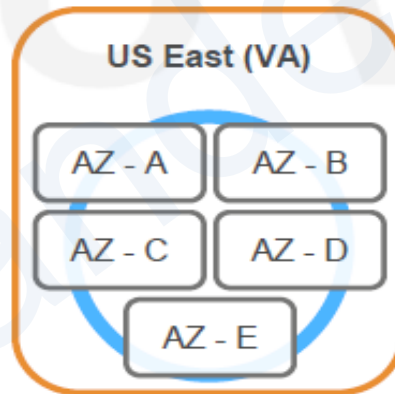
AWS Global Infrastructure

At least 2 AZs per region.

Examples:

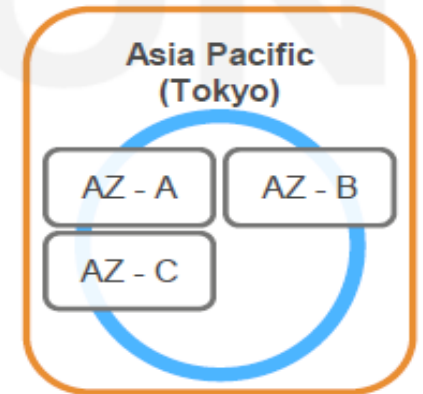
➤ US East (N. Virginia)

- us-east-1a
- us-east-1b
- us-east-1c
- us-east-1d
- us-east-1e



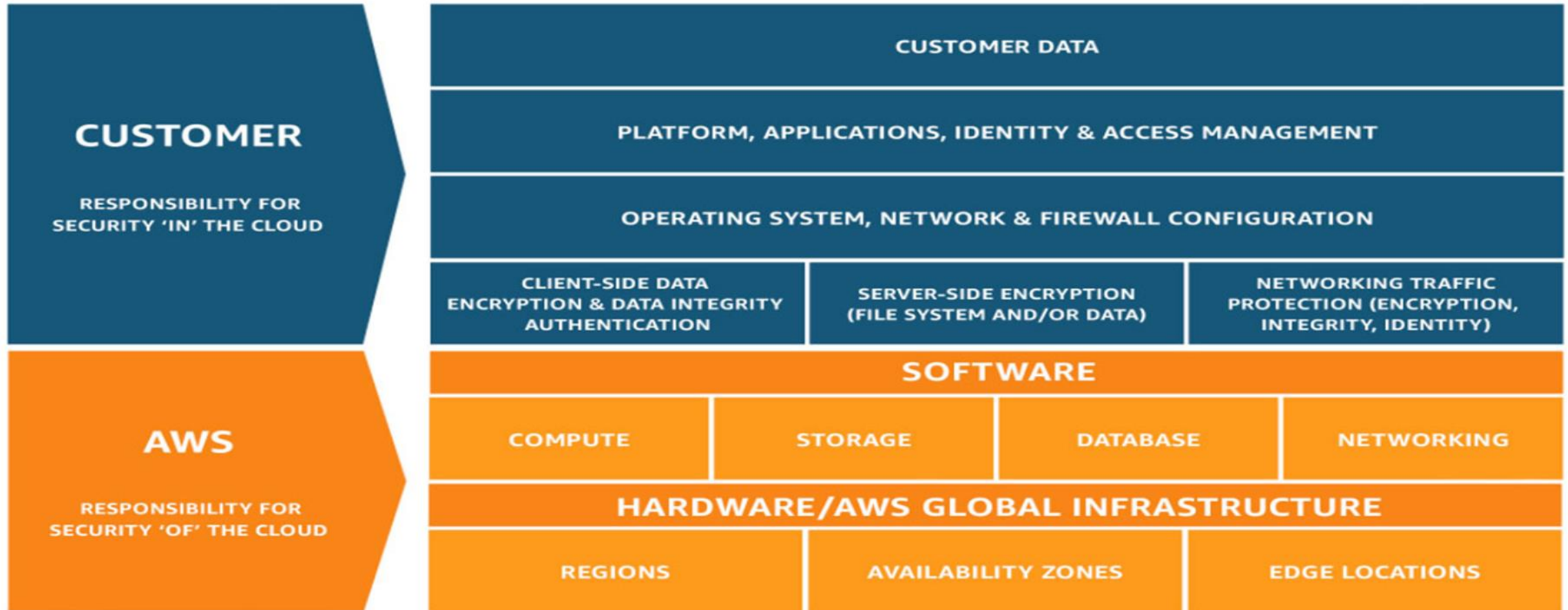
➤ Asia Pacific (Tokyo)

- ap-northeast-1a
- ap-northeast-1b
- ap-northeast-1c



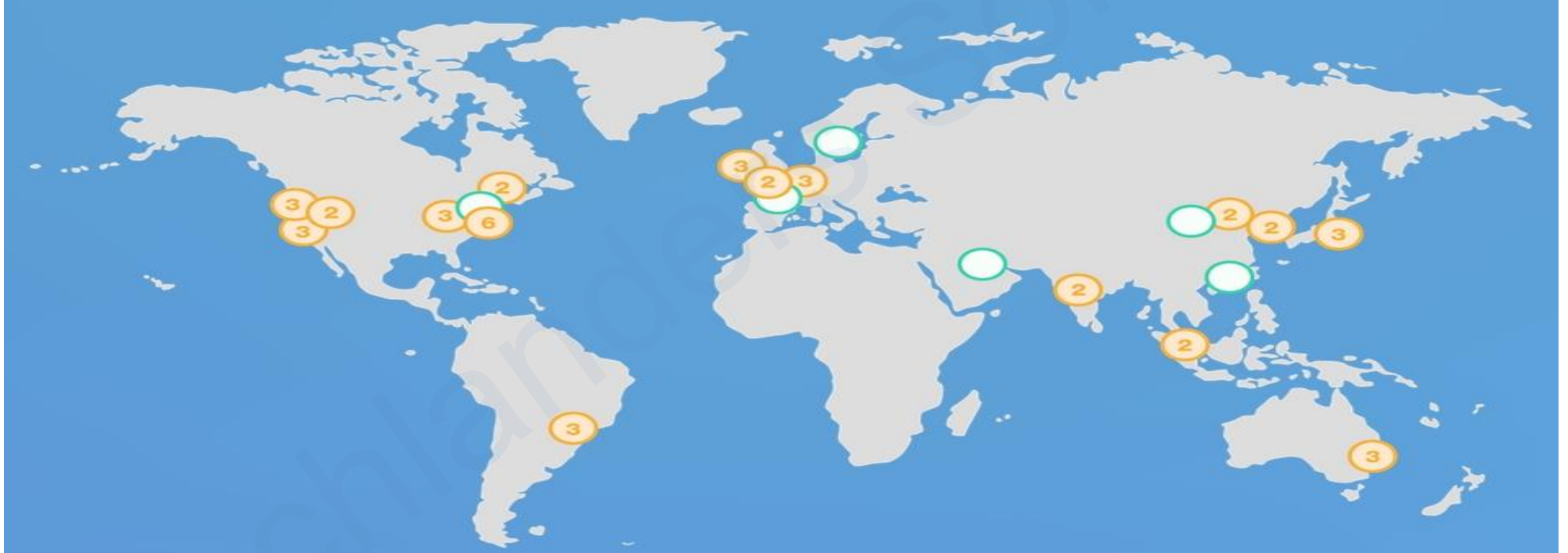
Note: Conceptual drawing only. The number of Availability Zones (AZ) may vary.

Shared Responsibility



AWS Global Infrastructure

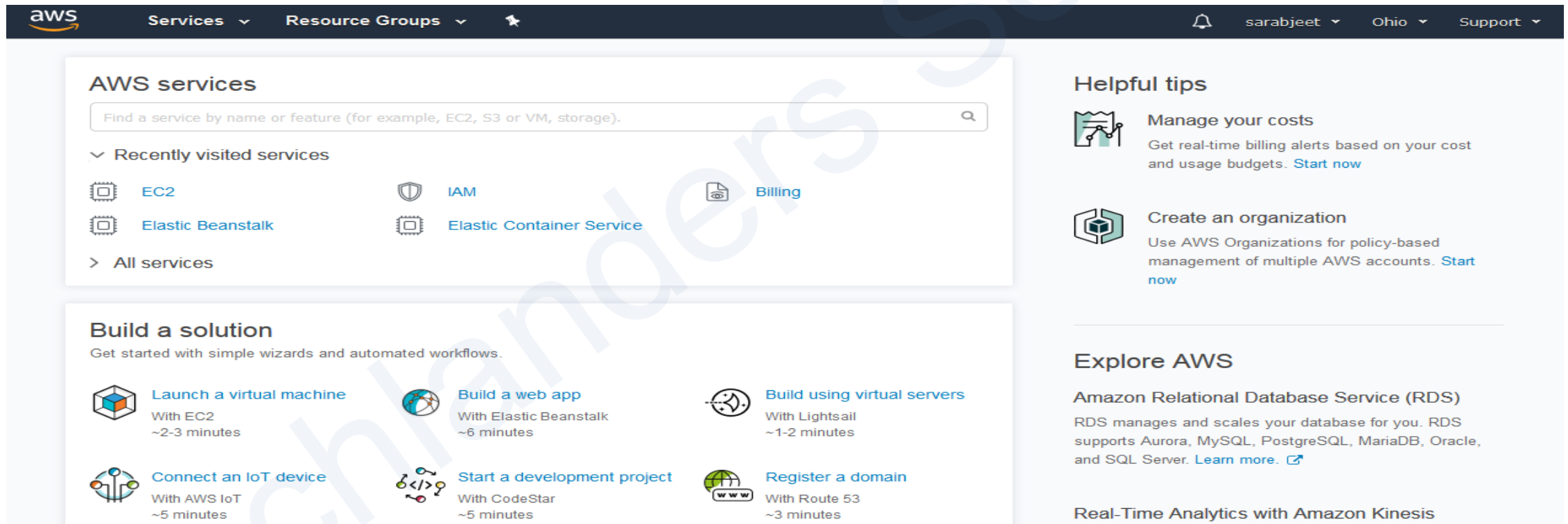
The AWS Cloud operates 44 Availability Zones within 16 geographic Regions around the world, with announced plans for 17 more Availability Zones and six more Regions in Bahrain, China, France, Hong Kong, Sweden, and a second AWS GovCloud Region in the US.



AWS Management Console

Simple and intuitive web-based user interface.

<https://console.aws.amazon.com/>



The screenshot displays the AWS Management Console interface. At the top is a dark navigation bar with the AWS logo, 'Services' and 'Resource Groups' dropdown menus, a star icon, and user information including a notification bell, the name 'sarabjeet', the region 'Ohio', and a 'Support' dropdown. Below the navigation bar, the main content area is divided into several sections. The 'AWS services' section features a search bar with the placeholder text 'Find a service by name or feature (for example, EC2, S3 or VM, storage)'. Below the search bar, there is a 'Recently visited services' section with a grid of service tiles: EC2, IAM, Billing, Elastic Beanstalk, and Elastic Container Service. A link for 'All services' is at the bottom of this section. The 'Build a solution' section is titled 'Get started with simple wizards and automated workflows.' and contains six tiles: 'Launch a virtual machine' (With EC2, ~2-3 minutes), 'Build a web app' (With Elastic Beanstalk, ~6 minutes), 'Build using virtual servers' (With Lightsail, ~1-2 minutes), 'Connect an IoT device' (With AWS IoT, ~5 minutes), 'Start a development project' (With CodeStar, ~5 minutes), and 'Register a domain' (With Route 53, ~3 minutes). On the right side, the 'Helpful tips' section includes two items: 'Manage your costs' (Get real-time billing alerts based on your cost and usage budgets. Start now) and 'Create an organization' (Use AWS Organizations for policy-based management of multiple AWS accounts. Start now). Below this is the 'Explore AWS' section, which highlights 'Amazon Relational Database Service (RDS)' and 'Real-Time Analytics with Amazon Kinesis'.

AWS services

Find a service by name or feature (for example, EC2, S3 or VM, storage).

▼ Recently visited services

- EC2
- IAM
- Billing
- Elastic Beanstalk
- Elastic Container Service

> All services

Build a solution

Get started with simple wizards and automated workflows.

- Launch a virtual machine**
With EC2
~2-3 minutes
- Build a web app**
With Elastic Beanstalk
~6 minutes
- Build using virtual servers**
With Lightsail
~1-2 minutes
- Connect an IoT device**
With AWS IoT
~5 minutes
- Start a development project**
With CodeStar
~5 minutes
- Register a domain**
With Route 53
~3 minutes

Helpful tips

- Manage your costs**
Get real-time billing alerts based on your cost and usage budgets. [Start now](#)
- Create an organization**
Use AWS Organizations for policy-based management of multiple AWS accounts. [Start now](#)

Explore AWS

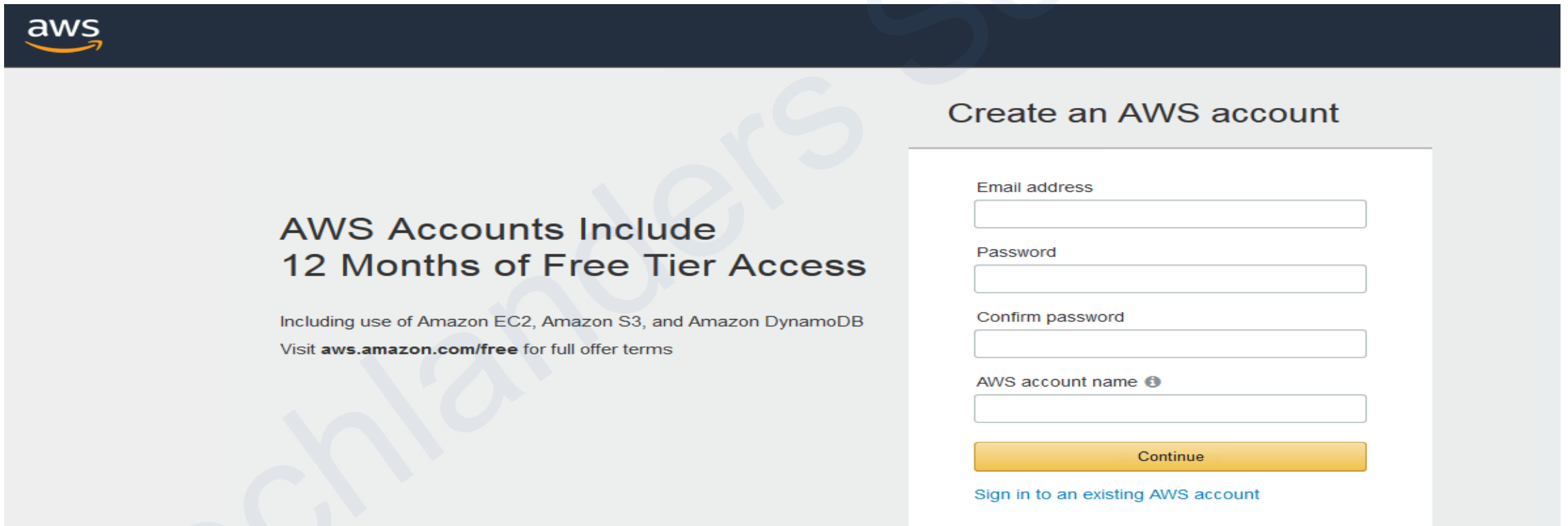
Amazon Relational Database Service (RDS)
RDS manages and scales your database for you. RDS supports Aurora, MySQL, PostgreSQL, MariaDB, Oracle, and SQL Server. [Learn more.](#)

Real-Time Analytics with Amazon Kinesis

LAB 1 : AWS Signup

Create a new account at

<https://portal.aws.amazon.com/billing/signup#/start>



The screenshot shows the AWS Signup page. On the left, there is a promotional message about the 12-month free tier. On the right, there is a form to create a new AWS account. The form includes fields for Email address, Password, Confirm password, and AWS account name, followed by a Continue button and a link to sign in to an existing account.

aws

Create an AWS account

**AWS Accounts Include
12 Months of Free Tier Access**

Including use of Amazon EC2, Amazon S3, and Amazon DynamoDB
Visit aws.amazon.com/free for full offer terms

Email address

Password

Confirm password

AWS account name ⓘ

Continue

[Sign in to an existing AWS account](#)

AWS SIGNUP

After creating the account

andhi Nagar

of suite, unit, building, floor, etc

Province or region

Code

Amazon Internet Services Pvt. Ltd. Customer

Customers with an India contact address are now required to register with Amazon Internet Service Private Ltd. (AISPL), the local seller for AWS infrastructure services in India.

Click here to indicate that you have read and agree to the terms of the [AISPL Customer Agreement](#)

Create Account and Continue

Payment Information

We use your payment information to verify your identity and only for usage in excess of the AWS Free Tier Limits. [We will not charge you for usage below the AWS Free Tier Limits.](#) For more information, see the [frequently asked questions](#).



As part of our card verification process we will charge INR 2 on your card when you click the "Secure Submit" button below. This will be refunded once your card has been validated. Your bank may take 3-5 business days to show the refund. Mastercard/Visa customers may be redirected to your bank website to authorize the charge.

Credit/Debit card number

Expiration date

10 ▼

2019 ▼

Cardholder's name

Select a Support Plan

AWS offers a selection of support plans to meet your needs. Choose the plan that best aligns with your AWS usage. [Learn more](#)



Basic Plan

Free

- Included with all accounts
- 24x7 self-service access to AWS resources
- For account and billing issues only
- Access to Personal Health Dashboard & Trusted Advisor



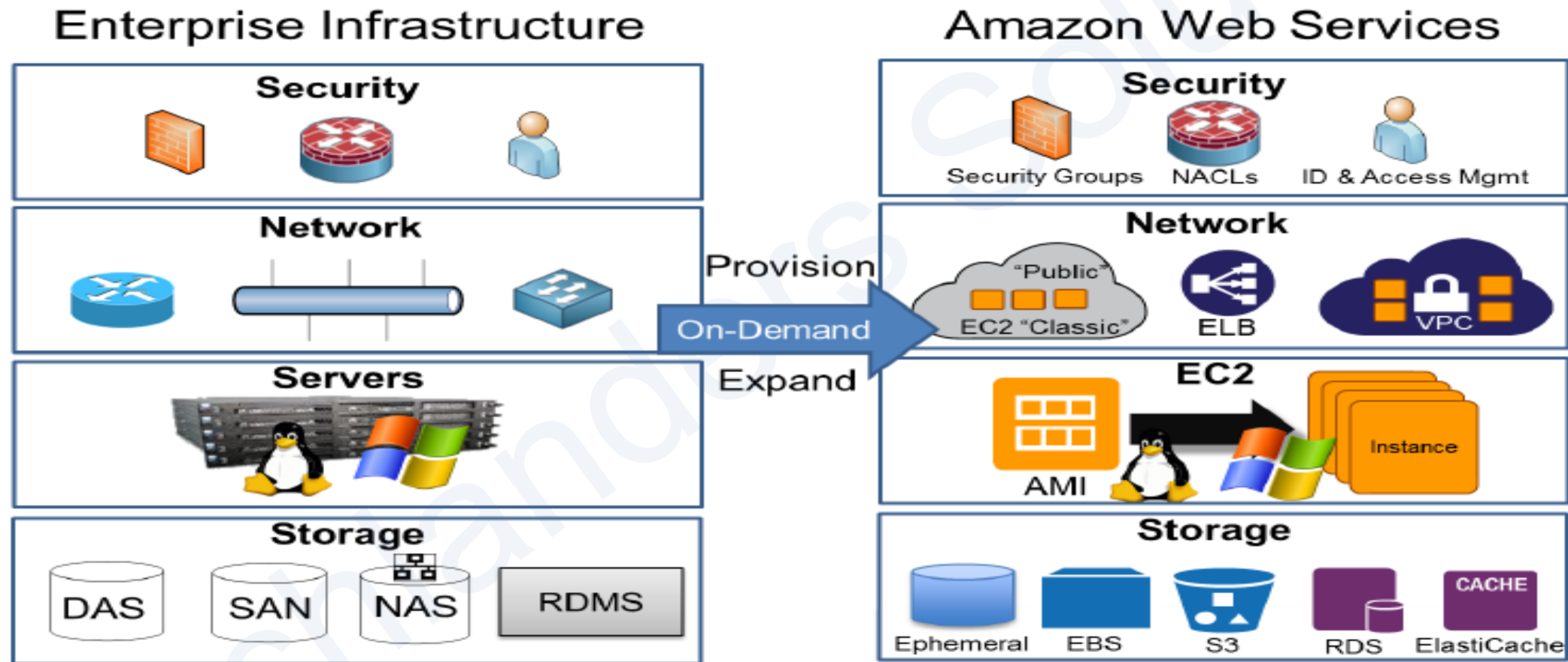
Developer Plan

From \$29/month

- For early adoption, testing and development
- Email access to AWS Support during business hours
- 1 primary contact can open an unlimited number of support cases
- 12-hour response time for nonproduction systems

Need Enterprise level support?

AWS Core Infrastructure Services



AWS Security

Physical Security:

24/7 trained security staff

AWS data centers in nondescript and undisclosed facilities

Two-factor authentication for authorized staff

Authorization for data center access

Multiple approval based change process



AWS Security

Hardware, Software, and Network :

Authentication and authorization in place

RBAC based access control mechanism

Firewall and other boundary devices

Security at Server level, Application level and Network level

AWS monitoring tools

Services to log AWS resources access



AWS Security



Amazon Resource Names (ARNs)

Amazon Resource Names (ARNs) uniquely identify AWS resources.

We require an ARN when you need to specify a resource unambiguously across all of AWS, such as in IAM policies, API calls etc.

ARN have a specific format:

arn:partition:service:region:account-id:resourcetype/resource

IAM user name

arn:aws:iam::123456789012:user/David


IAM instance id:

arn:aws:ec2:region:account-id:dedicated-host/host_id

Eg. arn:aws:ec2:us-east-1:123456789012:dedicated-host/h-12345678

AWS Access Credentials

AWS resources can be access using several authentication methods:

- IAM User-id / Password
 - Account ID/ AK (Access Key)/ SK (Security Key)
 - Certificates
 - Key pairs
- 

Resource Management Tools

AWS Management Console

AWS Console Mobile App (View resources)

AWS Command line interface

AWS Toolkit for PowerShell

AWS-Shell



AWS IAM

AWS Identity and Access Management

(IAM)

AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources for your users. You use IAM to control who can use your AWS resources (***authentication***) and what resources they can use and in what ways (***authorization***).

Shared access to your AWS account

Granular permissions

Secure access to AWS resources for applications that run on Amazon EC2

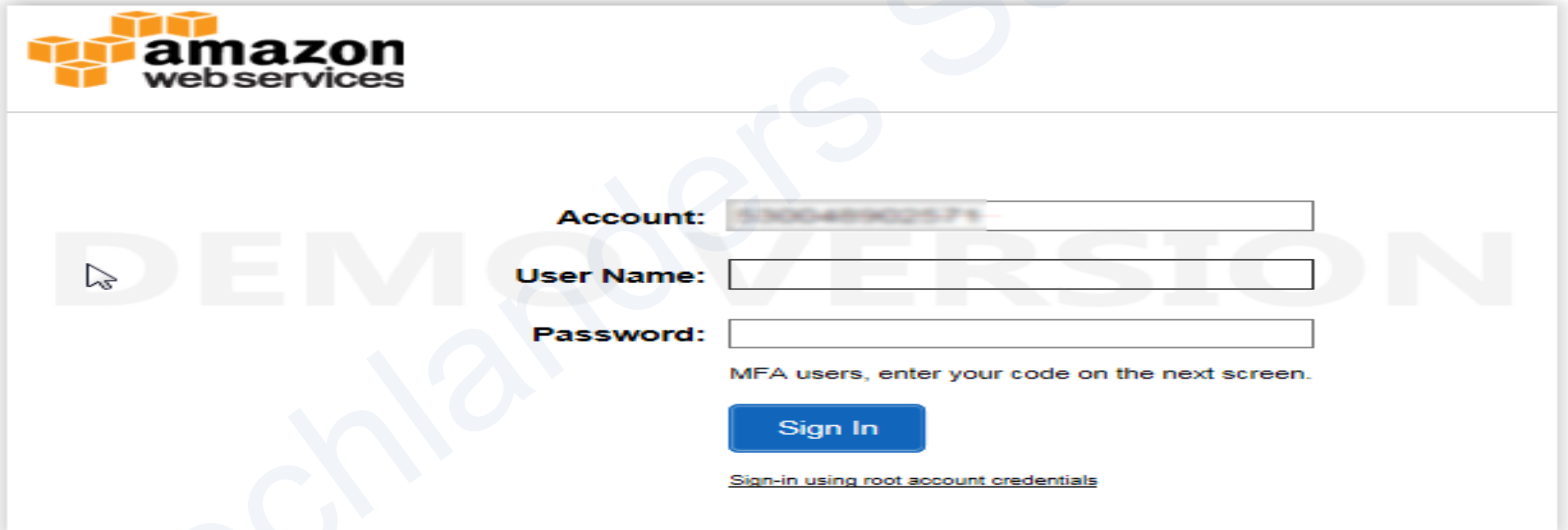
Integrated with many AWS services

Free to use



IAM - Users

You can create IAM users to distribute your work among team/users.
Different permissions can be given to IAM users



The screenshot shows the Amazon IAM console sign-in interface. At the top left is the Amazon Web Services logo. The main area contains three input fields: 'Account:' with a pre-filled value '111111111111', 'User Name:', and 'Password:'. Below these fields is a note for MFA users and a 'Sign In' button. A link for root account sign-in is at the bottom.

Account:

User Name:

Password:

MFA users, enter your code on the next screen.

[Sign In](#)

[Sign-in using root account credentials](#)

LAB 27 : IAM User

Create an IAM User

In this exercise, you will create an IAM user who can perform all administrative IAM functions. Then you will log in as that user so that you no longer need to use the root user login. Using the root user login only when explicitly required is a recommended security practice (along with adding MFA to your root user).


1. While logged in as the root user, create a new IAM user called user1.
2. Provide Access type -> Programmatic access and Management console access
3. Set a console password and click next
4. Logout from root and Use the customized sign-in link to sign in as Administrator
5. Try to login and check if you are able to access the s3 buckets in new user.
6. Were to able to see? What about other services?

IAM - Groups

Users are part of IAM groups

Common policies can be applied on groups

▼ Summary

Group ARN:	arn:aws:iam::242959489011:group/read_group 
Users (in this group):	2
Path:	/
Creation Time:	2018-05-02 22:21 UTC+0530

Users

Permissions

Access Advisor


This view shows all users in this group: **2 Users**

User	Actions
 gagandeep	Remove User from Group
 Tejas	Remove User from Group

LAB 26 : IAM Group

Create an IAM Group

In this exercise, you will create a group for all IAM administrator users and assign the proper permissions to the new group. This will allow you to avoid assigning policies directly to a user later in these exercises.

1. Log in as the root user.
 2. Go to **Security, Identity & Compliance** → **IAM** → **Groups**
 3. Create an IAM group provide name Elevated_Users
 4. Don't select any policy and click create.
 5. Add user created to the group
- 

AWS

Compute Services

Virtual Machines

What are the bare minimum infrastructure requirements for any application?

Hardware (RAM, Processor, Processor type, HBA's, etc.)

OS Disk & Data Disk

OS Images

Network

Security (Firewall, Networking, Access Mechanism)

Credentials



AWS Elastic Compute Cloud

Amazon **EC2** stands for **E**lastic **C**ompute **C**loud, and is the Primary AWS web service.

Provides Resizable compute capacity

Reduces the time required to obtain and boot new server instances to minutes

There are two key concepts to Launch instances in AWS:

- Instance Type**

- AMI**

EC2 Facts:

Scale capacity as your computing requirements change

Pay only for capacity that you actually use

Choose Linux or Windows OS as per need.

Deploy across AWS Regions and Availability Zones for reliability/HA

Only X86 based OS supported. So platform specific OS are not supported.

Instance Types (EC2)

The instance type defines the different virtual hardware Models (sizes) supporting an Amazon EC2 instance.

There are dozens of instance types available, varying in the following dimensions:

- Virtual CPUs (vCPUs)

- Memory

- Storage (size and type)

- Network performance

- Graphical Processing

Instance types are grouped into families based on the ratio of these values to each other.

Instance Types (EC2)

EC2 instance types are optimized for different use cases and come in multiple sizes. This allows you to optimally scale resources to your workload requirements.

AWS uses Intel® Xeon® processors for EC2 instances, providing customers with high performance and value.

Consider the following when choosing your instances: Core count, memory size, storage size and type, network performance, and CPU technologies.

Hurry Up and Go Idle - A larger compute instance can save you time and money, therefore paying more per hour for a shorter amount of time can be less expensive.

Instance Types with AWS

General
purpose



Compute
optimized



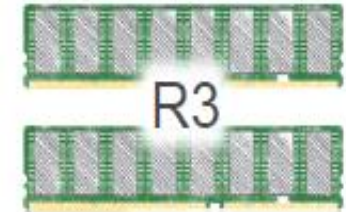
Storage and IO
optimized



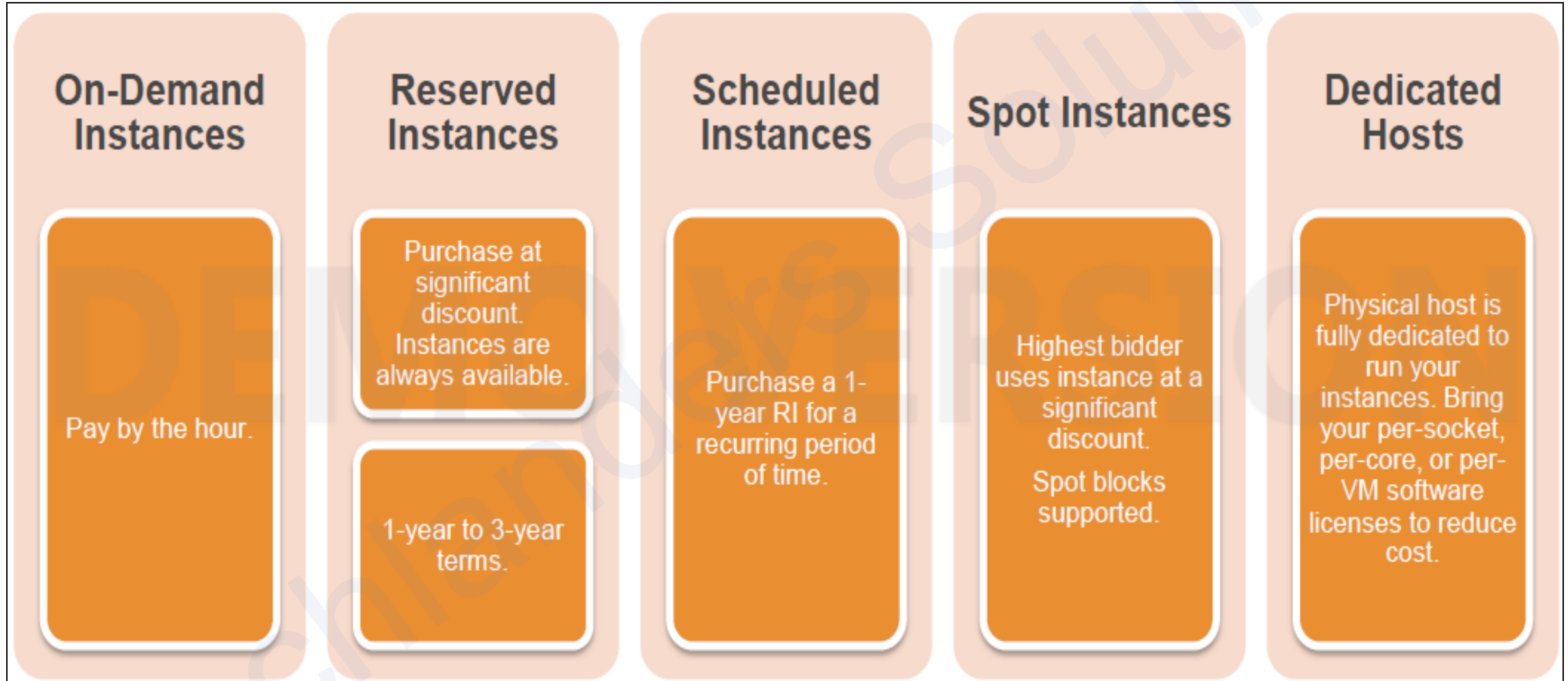
GPU
enabled



Memory
optimized



EC2 Pricing Models



AMI's

Stands for **Amazon Machine Image**

AMI is a template for the root volume for the instance (example: an OS image, a webserver, an application server etc.)

It also includes the launch permissions that control which AWS accounts can use the AMI to launch instances.

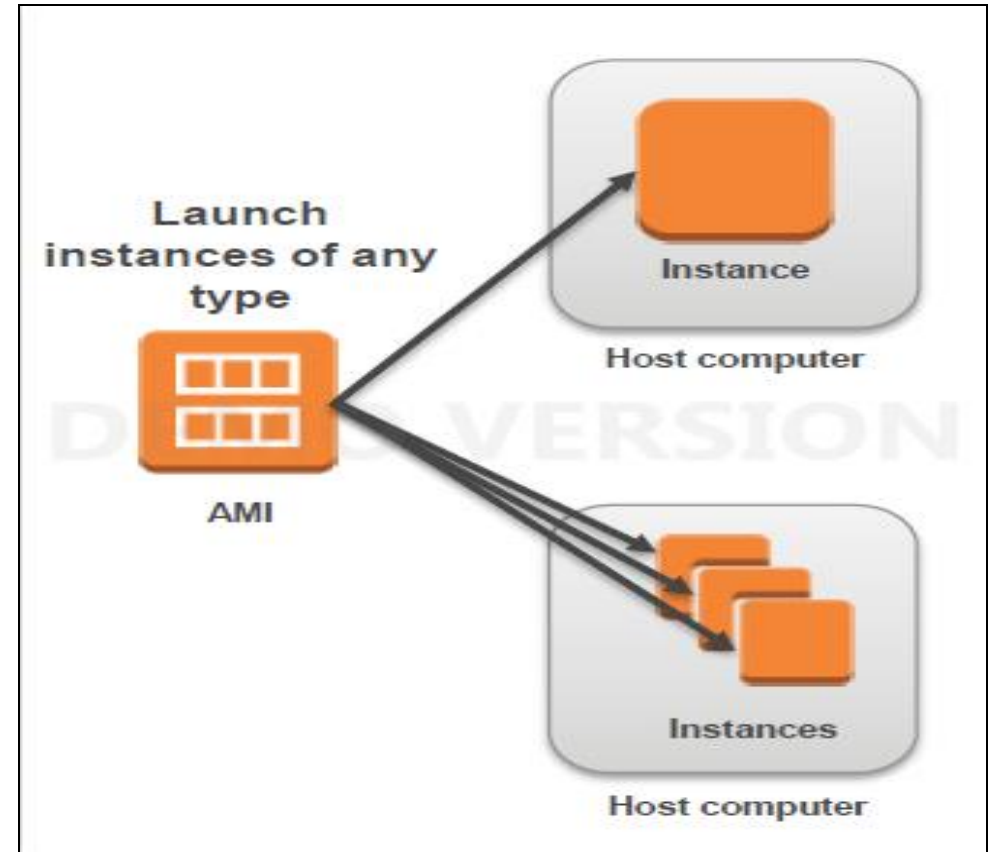
All AMIs are based on x86 OSs, either Linux or Windows.

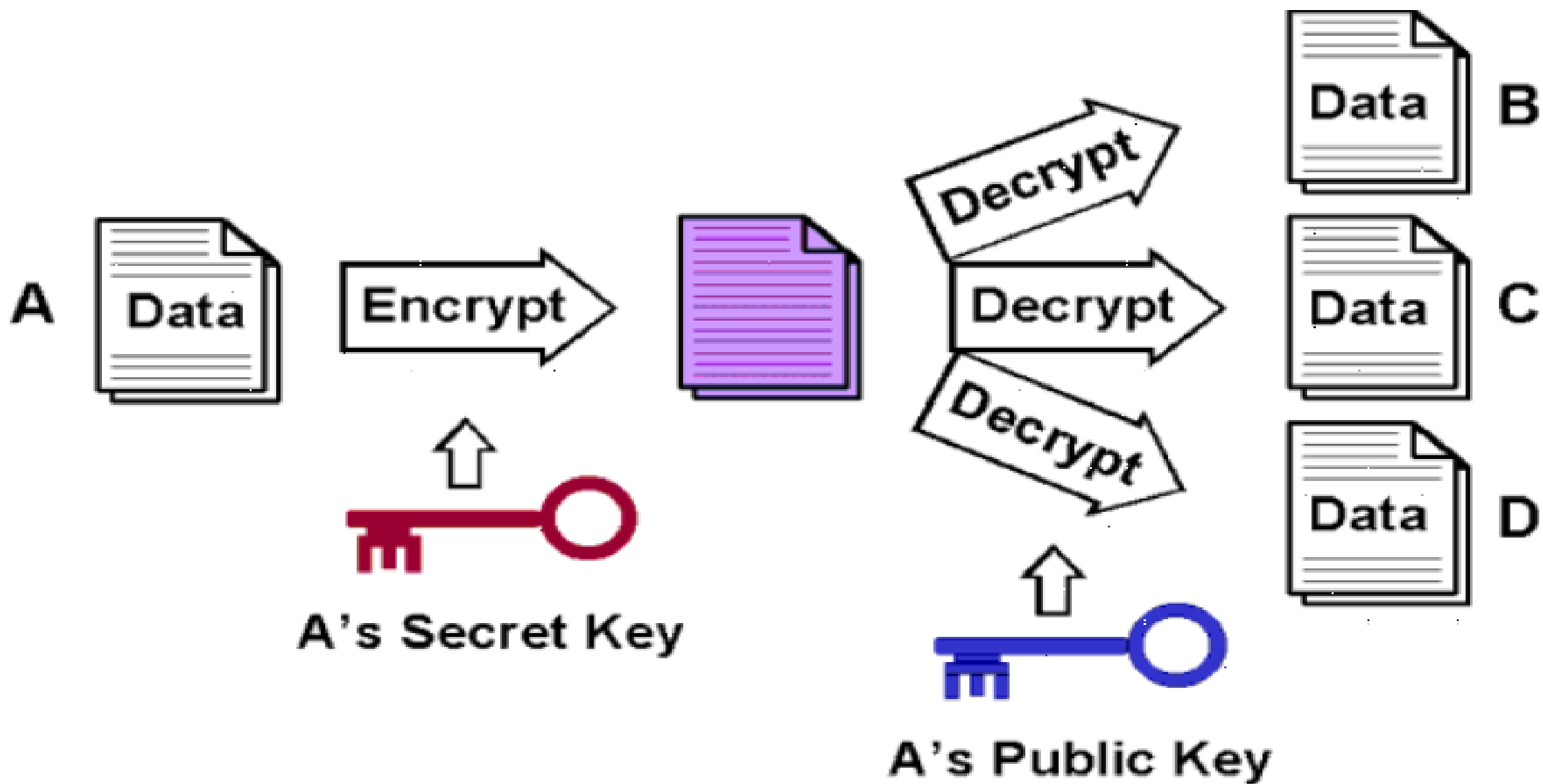


Instances and AMI's

Select an AMI based on:

- Region
- Operating system
- Architecture (32-bit or 64-bit)
- Launch permissions
- Storage for the root device
- Virtualization Type





Credentials

Used to access an Instance.


Amazon EC2 uses public-key cryptography to encrypt and decrypt login information.

Public-key cryptography uses a public key to encrypt a piece of data and an associated private key to decrypt the data.

These two keys together are called a *key pair*.

AWS stores the public key, and the private key is kept by the customer. The private key is essential to acquiring secure access to an instance for the first time.

When launching a Windows instance, Amazon EC2 generates a random password for the local administrator account and encrypts the password using the public key.



Launching EC2 Instance


Procedure to launch EC2 instance in minutes:

Determine the AWS Region in which you want to launch the Amazon EC2 instance.

Launch an Amazon EC2 instance from a pre-configured Amazon Machine Image (AMI).

Choose an instance type based on CPU, memory, storage, and network requirements.

Configure network, security groups, storage volume, tags, and key pair; or directly launch instance post selecting AMI and Instance type.



LAB 3 : Create an EC2 Linux Instance

Console Access: <https://console.aws.amazon.com/>

Observe different available AMIs and Instance types

Exercise:

Create a Linux Operating System using mandatory configurations:

Instance Type : **T2.Micro**

AMI : **Amazon Linux 2 AMI**

Credentials: **Create/download your own key pair and use same.**

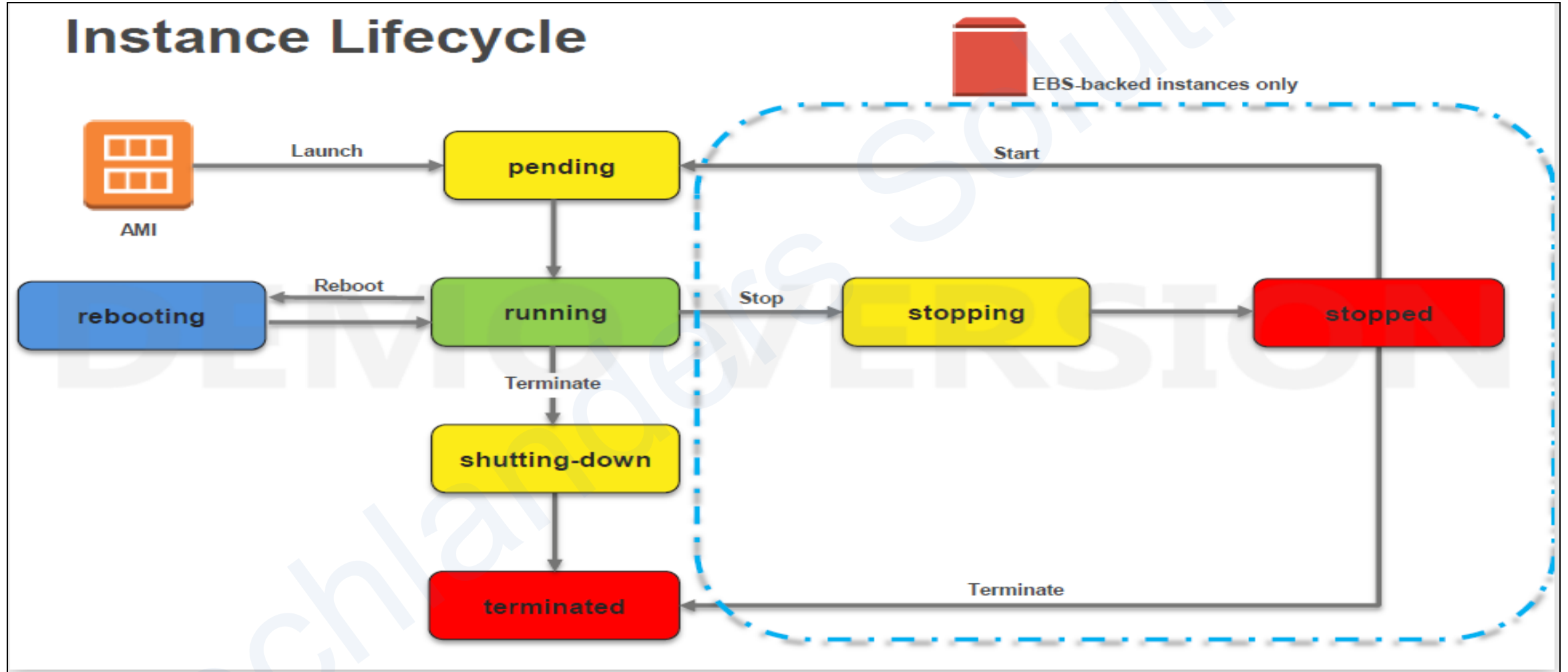
Observe the instance details and navigate through the diff tabs

Reboot, Stop, Start your server

Access your instance using above downloaded key pair using different platforms i.e ,
instance connect , using putty , cli.

Note: (For the first instance take the default parameters wherever inputs required)

Instance Lifecycle




Network part

Private IP: Alike traditional VMs/hosts, each EC2 instance must at least belong to one network (VPC in AWS) and must have at least one Private IP from that network/subnet.

Public IP: A launched instance may also have a public IP address assigned. This IP address is assigned from the addresses reserved by AWS and cannot be specified. This IP address is unique on the Internet, persists only while the instance is running, and cannot be transferred to another instance.

Private/Public Domain Name System (DNS): When you launch an instance, AWS creates one private and one Public DNS name that can be used to access the instance. This DNS name is generated automatically and cannot be specified by the customer. This DNS name persists only while the instance is running and cannot be transferred to another instance.

Elastic IP (EIP): An elastic IP address is an address unique on the Internet that you reserve independently and associate with an Amazon EC2 instance.



EC2 Security – Security Group

Virtual Firewall Protection

AWS allows you to control traffic **in** and **out** of your instances through virtual firewalls called *security groups*.

Security groups allow you to control traffic based on *port, protocol, and source(inbound)/destination(outbound)*.

Security groups are associated with instances when they are launched. **Every instance must have at least one security group.** Though they can have more.

A security group is default deny.



Amazon Server Storage

Amazon EBS (Elastic Block Store)

- Data stored on an Amazon EBS volume can persist independently of the life of the instance.
- Persistent **block-level storage volumes** for use with Amazon EC2 instances.
- 99% used volume type in AWS.

Amazon EC2 Instance Store

- Like Swap volumes attached to your server for faster processing.
- Data stored on a local instance store persists only as long as the instance is alive.
- Storage is ephemeral.
- Older volume type, used exceptionally in some instances i.e DB, High processing systems.

AWS Tags

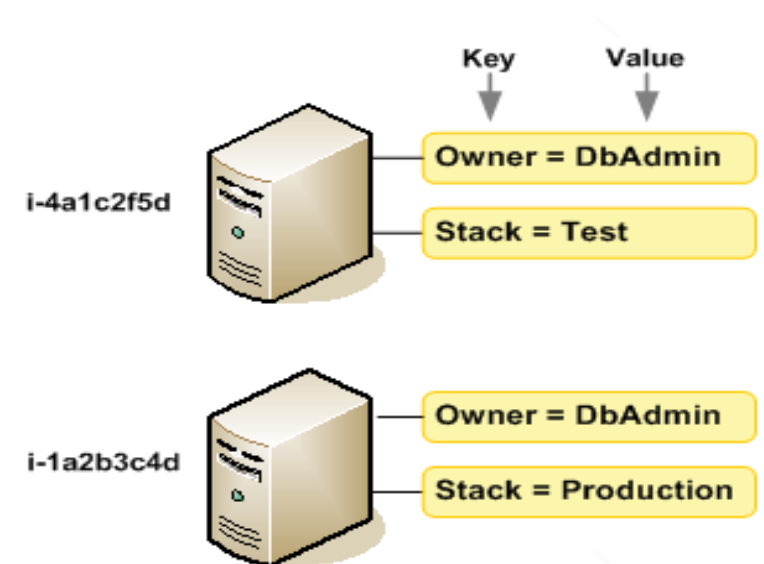
Tags helps you manage your instances, images, and other Amazon EC2 resources.

Its your own metadata to each resource in the form of *tags*.

Each tag consists of a *key* and an optional *value*, both of which you define.

It enable you to categorize your AWS resources in different ways, for example, by purpose, owner, or environment.

You can organize your billing information based on resources that have the same tag key values.



AWS Tags

Maximum number of tags per resource—50

Tag keys and values are case-sensitive.

You can't terminate, stop, or delete a resource based solely on its tags; you must specify the resource identifier : ARN's

Not all resources supports tags. Pls check the supported list from AWS website.



EC2 Metadata

AWS Instance Metadata –

Instance metadata is data/information about your instance.

An HTTP call to **`http://169.254.169.254/latest/meta-data/`** will return the top node of the instance metadata tree and will return the information about your instance e.g. Public IP, DNS name, OS type etc..

For Linux systems:

curl <http://169.254.169.254/latest/meta-data/>

Dynamic Instance Identity metadata:

<http://169.254.169.254/latest/dynamic/instance-identity/>

LAB 4 : Create Windows Instance

Exercise:

Create a Windows Operating System using custom configurations:

Instance Type : **T2.Micro**

AMI : **Microsoft Windows Server 2019 Base**

Credentials: **Use existing key from Lab 3 or if not create a new one.**

Assign tags, **Key = Name, Value= Server-Name**

Assign tags, **Key = Owner, Value= Your name**

Add inbound port of **RDP**

Access your instance using Administrator password, which will be retrieved using your downloaded key pair using mstsc or dns

AWS CLI

AWS CLI is a command based utility to manage AWS resources

The primary distribution method for the AWS CLI on Linux, Windows, and macOS is pip, a package manager for Python that provides an easy way to install, upgrade, and remove Python packages and their dependencies

<http://docs.aws.amazon.com/cli/latest/userguide/installing.html>

Requirements

- Python 2 version 2.6.5+ or Python 3 version 3.3+

- Windows, Linux, macOS, or Unix

- Pip package should be present (else install python-pip)

Install AWSCLI: **pip install awscli --upgrade --user**

For Windows, directly download the Windows installer from CLI webpage

LAB 2 : Install AWS CLI

Lets install an AWSCLI

<https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>

aws --version

aws help

aws ec2 help / aws s3 help / aws <anysubcommand> help

Configure your default keys and region:

```
root@ip-172-31-28-145:~# aws configure
AWS Access Key ID [None]: #####
AWS Secret Access Key [None]: #####
Default region name [None]: us-west-2
Default output format [None]:
root@ip-172-31-28-145:~#
```

LAB 2 : Install AWS CLI

Watch Credentials and configurations in .aws directory under current directory being created/updated with aws configure:

```
C:\Users\gd>dir .aws
Volume in drive C is System
Volume Serial Number is 5205-3B4A
Directory of C:\Users\CQBJ1454\.aws
2018-02-11  19:13    <DIR>        .
2018-02-11  19:13    <DIR>        ..
2018-02-11  19:29                48 config
2018-05-05  09:03               119 credentials
                2 File(s)        167 bytes
                2 Dir(s)  452,272,128 bytes free
C:\Users\gd>
```


LAB 6 : Creating server using CLI

Check the details for all running instances using CLI

```
aws ec2 describe-instances | grep -i instanceid
```

Creation of an AWS Instance using CLI:

```
aws ec2 run-instances --image-id ami-033b95fb8079dc481 --instance-type t2.micro  
--key-name virginia-key
```

```
aws ec2 describe-instances | grep -i instanceid
```

```
aws ec2 stop-instances --instance-ids i-052af6a166198df10
```

```
aws ec2 terminate-instances --instance-ids i-052af6a166198df10
```

